暨南大學
JINAN UNIVERSITY

Project Report

(First semester for Year 2024-2025)

Intelligent Flight Simulation System

Course Name:	Human Computer Interaction
Course Type:	Optional
Student Name:	段瑞豪
Student ID:	2022180163
College:	International School
Department:	
Major:	Computer Science and Technology
Professor:	龙锦益

December 30th, 2024

1. Introduction

This report documents the final phase of the project, focusing on the prototype implementation. The primary aim of this phase is to demonstrate the software to the instructor, showcasing its functionality and interface, and to submit the fully commented source code. The demonstration will include specific use cases that highlight the capabilities of the software, ensuring a comprehensive understanding of its core features and performance.

2. Project Deliverables

The project deliverables consist of three key components: a live demonstration, specific use case evaluations, and the submission of the source code. Each element is designed to validate and present the project's functionality effectively.

2.1. Demonstration to the Instructor

The live demonstration is a critical part of this phase. During the demo, the software will be presented in a "ready-to-run" state, ensuring it is fully functional and accessible for evaluation. A sequence of predefined use cases has been prepared to showcase the software's key features systematically. These use cases are intended to guide the instructor through the interface and functionality, emphasizing the user experience and technical capabilities.

2.2. Use Cases and Test Scenarios

Specific use cases have been developed to illustrate the software's primary features and to test its performance under various conditions. These include the following:

Use Case 1: eVTOL Flight Simulation

The first use case demonstrates the simulation of eVTOL flight dynamics. It includes takeoff, forward motion, and landing. The user can adjust parameters such as thrust and weight via the user interface sliders, which dynamically affect the eVTOL's behavior. Observations will include hovering stability and responsiveness to parameter changes, providing insights into the accuracy of the implemented flight physics.

Use Case 2: Landing Pad Detection

This use case involves the detection of landing pads using computer vision techniques. A scene generated in Unity features a landing pad, which is processed using Python and OpenCV. The detection process will be illustrated in real-time, with visual feedback provided in Unity to confirm whether the landing pad has been successfully identified. This highlights the integration of machine learning tools with the Unity simulation environment.

2.3. Commented Source Code Submission

The source code for the project has been thoroughly commented to provide clarity and insight into the underlying algorithms and logic. It includes modules for flight physics, landing pad detection, user interface management, and the integration of Python with Unity. The commented code is organized to ensure that each functionality is easily identifiable and comprehensible.

3. Preparation for the Demo

3.1. Technical Setup

The technical setup for the demonstration has been meticulously prepared. The software is installed on a machine optimized for performance and compatibility. Unity project files and Python scripts have been tested to ensure seamless integration, and all necessary dependencies, including OpenCV, have been installed and verified. This preparation guarantees a smooth and uninterrupted demonstration.

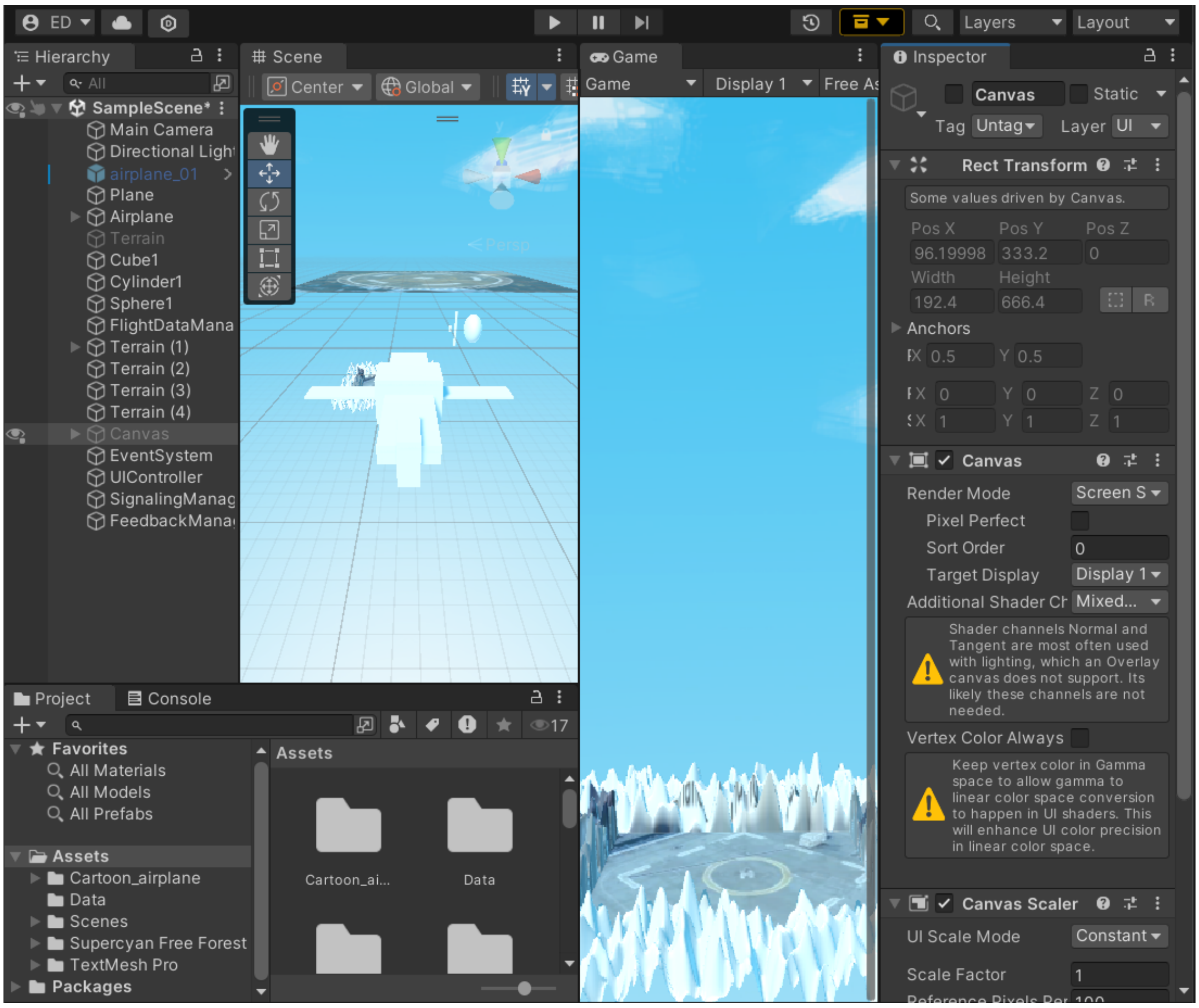
3.2. Printout of Use Cases

To aid the demonstration, a detailed printout of the use cases has been prepared. This document includes step-by-step instructions for each scenario, ensuring the instructor can follow the demonstration easily. The printout serves as a reference to validate the software's functionality against the defined requirements.

Use Case 1: eVTOL Flight Simulation

The first use case focuses on simulating the flight dynamics of an electric vertical takeoff and landing (eVTOL) aircraft. The simulation begins with the user interacting with a custom-designed interface in Unity, where parameters such as thrust and weight can be adjusted using intuitive sliders. By varying these inputs, the user can observe corresponding changes in the eVTOL's behavior. For example, increasing the thrust results in faster forward motion, while increasing the weight might cause the aircraft to struggle with maintaining altitude.

The takeoff phase is demonstrated by applying sufficient thrust to overcome the simulated weight and gravitational forces. Once airborne, the eVTOL hovers at a stable altitude, showcasing the implemented flight physics. Forward motion is initiated by adjusting the thrust further, demonstrating the aircraft's ability to transition smoothly between hovering and cruising states. Finally, the descent and landing phases are carefully controlled to illustrate the system's ability to manage altitude changes dynamically. Throughout this use case, the software's responsiveness to real-time user inputs is highlighted, validating the accuracy and reliability of the flight dynamics model.



Some values driven by Canvas.

Pos X 96.19998 Pos Y 333.2 Pos Z 0
Width 192.4 Height 666.4

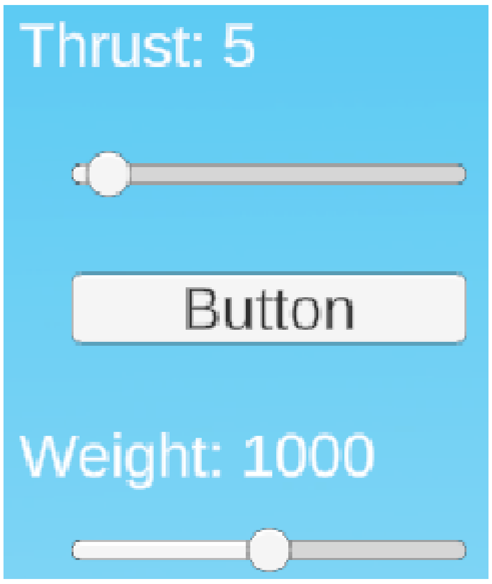
Render Mode: Screen S
Sort Order: 0
Target Display: Display 1
Additional Shader Ch: Mixed...

Shader channels Normal and Tangent are most often used with lighting, which an Overlay canvas does not support. Its likely these channels are not needed.

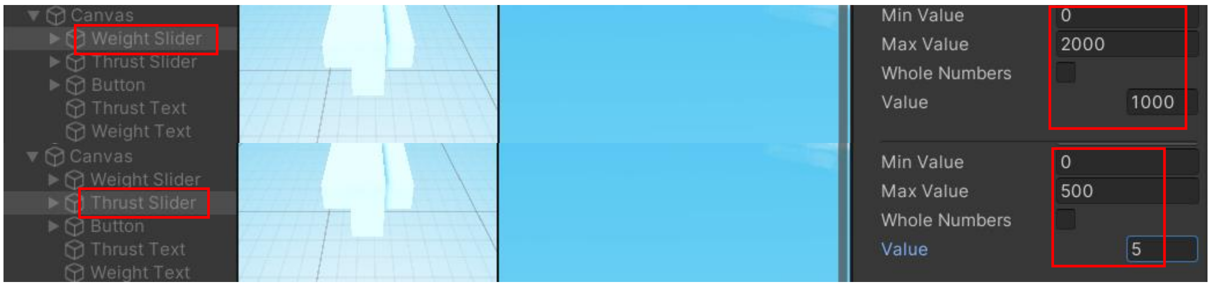
Keep vertex color in Gamma space to allow gamma to linear color space conversion to happen in UI shaders. This will enhance UI color precision in linear color space.

UI Scale Mode: Constant
Scale Factor: 1

Figure: Overview of the project in Unity



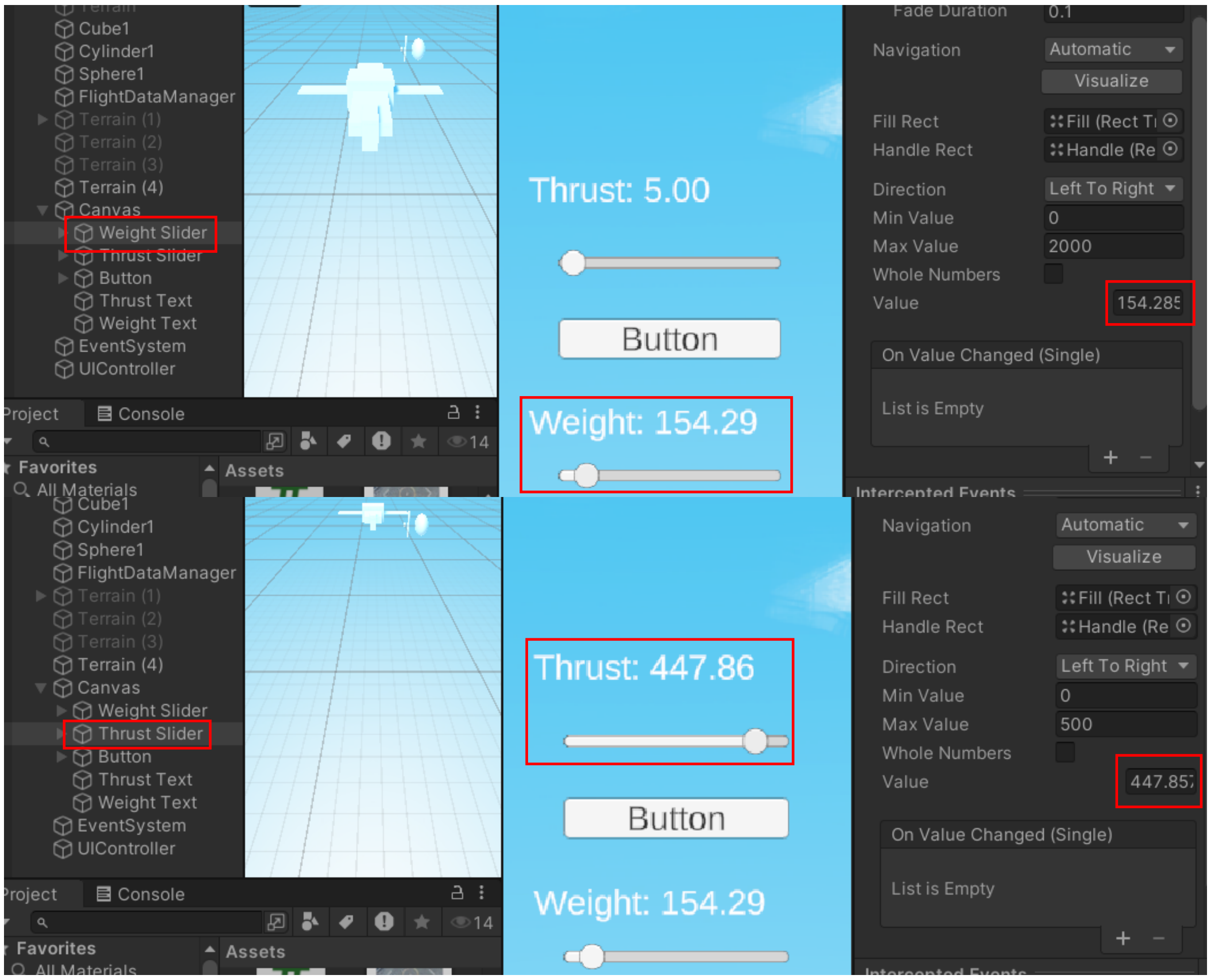
Figure： Parameters such as thrust and weight can be adjusted using intuitive sliders



Min Value 0
Max Value 2000
Whole Numbers
Value 1000

Min Value 0
Max Value 500
Whole Numbers
Value 5

Figure: Minimum, maximum and initial values set for the weight and thrust



Thrust: 5.00

Weight: 154.29

Fade Duration 0.1
Navigation Automatic
Visualize
Fill Rect Fill (Rect Tr
Handle Rect Handle (Re
Direction Left To Right
Min Value 0
Max Value 2000
Whole Numbers
Value 154.285
On Value Changed (Single)
List is Empty
Intercepted Events

Thrust: 447.86

Weight: 154.29

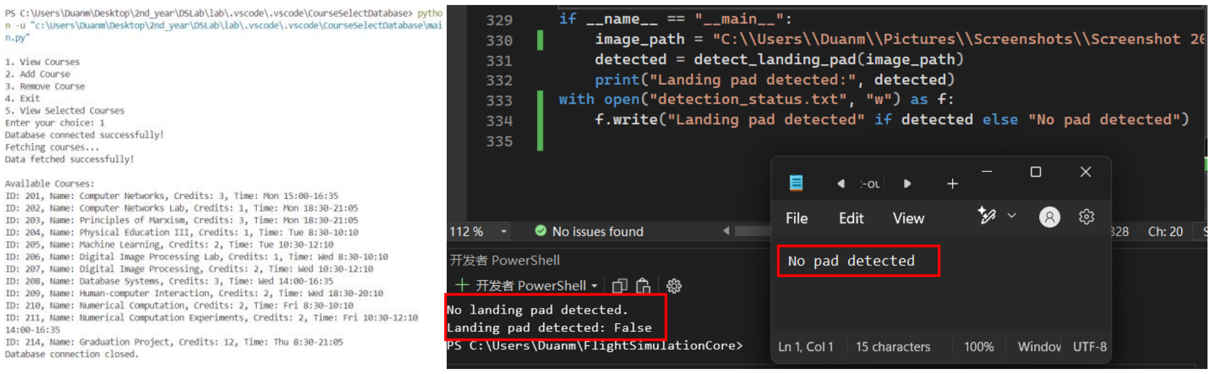
Navigation Automatic
Visualize
Fill Rect Fill (Rect Tr
Handle Rect Handle (Re
Direction Left To Right
Min Value 0
Max Value 500
Whole Numbers
Value 447.857
On Value Changed (Single)
List is Empty
Intercepted Events

Figure: The higher the thrust + the lower the weight, gives us a faster travelling plane.

Use Case 2: Landing Pad Detection

The second use case explores the integration of computer vision to detect landing pads within the simulation environment. A scene generated in Unity features a clearly marked landing pad with identifiable circular markings. Using OpenCV, the Python script processes frames captured from the simulation to identify these visual cues. The detection process involves converting the captured image to grayscale, applying the Hough Circle Transform to locate circular shapes, and marking detected landing pads visually.

The results of the detection are communicated back to Unity, providing real-time feedback on the detection status. When a landing pad is identified, the interface displays a confirmation message, "Landing pad detected," and highlights the detected area in the simulation view. This real-time interaction between Python and Unity underscores the system's ability to integrate external computer vision capabilities seamlessly. The robustness of the detection mechanism is tested under various conditions, such as different lighting scenarios and partial obstructions of the landing pad, to ensure reliability.



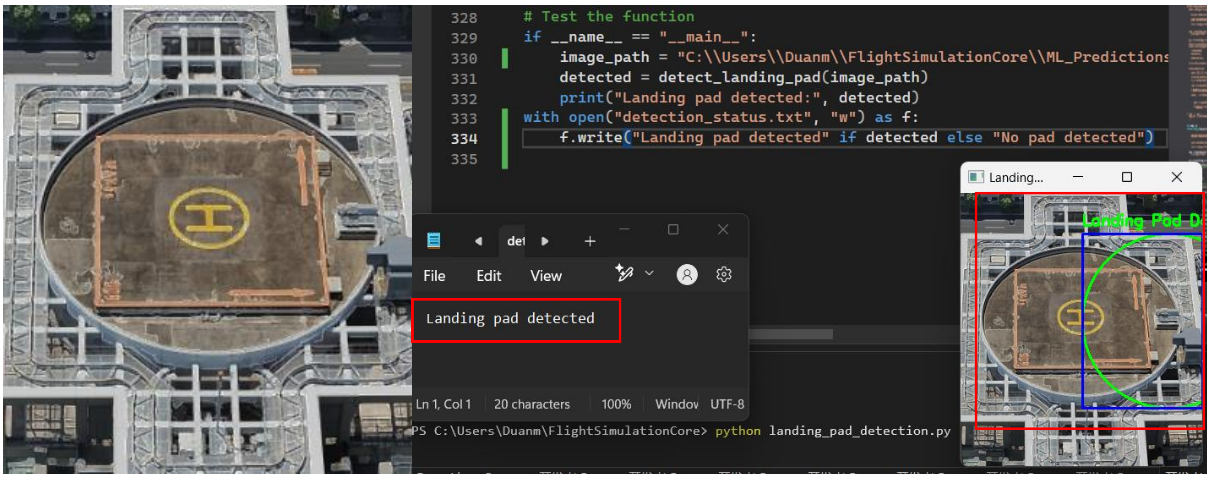
```
PS C:\Users\Duanm\Desktop\2nd_year\DSLab\lab\.vscode\.vscode\CourseSelectDatabase> pytho
n -u "c:\Users\Duanm\Desktop\2nd_year\DSLab\lab\.vscode\.vscode\CourseSelectDatabase\mai
n.py"

1. View Courses
2. Add Course
3. Remove Course
4. Exit
5. View Selected Courses
Enter your choice: 1
Database connected successfully!
Fetching courses...
Data fetched successfully!

Available Courses:
ID: 201, Name: Computer Networks, Credits: 3, Time: Mon 15:00-16:35
ID: 202, Name: Computer Networks Lab, Credits: 1, Time: Mon 18:30-21:05
ID: 203, Name: Principles of Marxism, Credits: 3, Time: Mon 18:30-21:05
ID: 204, Name: Physical Education III, Credits: 1, Time: Tue 8:30-10:10
ID: 205, Name: Machine Learning, Credits: 2, Time: Tue 10:30-12:10
ID: 206, Name: Digital Image Processing Lab, Credits: 1, Time: Wed 8:30-10:10
ID: 207, Name: Digital Image Processing, Credits: 2, Time: Wed 10:30-12:10
ID: 208, Name: Database Systems, Credits: 3, Time: Wed 14:00-16:35
ID: 209, Name: Human-computer Interaction, Credits: 2, Time: Wed 18:30-20:10
ID: 210, Name: Numerical Computation, Credits: 2, Time: Fri 8:30-10:10
ID: 211, Name: Numerical Computation Experiments, Credits: 2, Time: Fri 10:30-12:10
14:00-16:35
ID: 214, Name: Graduation Project, Credits: 12, Time: Thu 8:30-21:05
Database connection closed.
```

```python
if __name__ == "__main__":
    image_path = "C:\\Users\\Duanm\\Pictures\\Screenshots\\Screenshot 20
    detected = detect_landing_pad(image_path)
    print("Landing pad detected:", detected)
with open("detection_status.txt", "w") as f:
    f.write("Landing pad detected" if detected else "No pad detected")
```

```
No landing pad detected.
Landing pad detected: False
PS C:\Users\Duanm\FlightSimulationCore>
```

```
No pad detected
```

Figure: Using a random invalid picture (left), we were able to prove that it is not a landing pad



```python
# Test the function
if __name__ == "__main__":
    image_path = "C:\\Users\\Duanm\\FlightSimulationCore\\ML_Predictions
    detected = detect_landing_pad(image_path)
    print("Landing pad detected:", detected)
with open("detection_status.txt", "w") as f:
    f.write("Landing pad detected" if detected else "No pad detected")
```

```
Landing pad detected
```

```
PS C:\Users\Duanm\FlightSimulationCore> python landing_pad_detection.py
```

Figure: A real world landing pad (left) was able to be detected as a genuine one

4. Conclusion

The prototype implementation phase represents the culmination of the project, integrating multiple components into a unified and functional system. Through the live demonstration, the instructor will gain a clear understanding of the software's capabilities, while the submitted source code provides a detailed view of its technical structure. This phase validates the effectiveness and innovation of the project, showcasing its potential for real-world application.